Всероссийская олимпиада школьников
Муниципальный этап
9-11 классы

Задача №1. Стабильная машина

Ученый-исследователь в таблицу последовательно записывал результаты испытаний работы новой машины. Результатом каждого испытания является некоторое целое положительное число. Машина работает тем стабильнее, чем меньше перепад значений соседних испытаний.

Требуется: написать программу, определяющую по предложенным данным максимальный перепад значений соседних испытаний, и вывести их номера. Если таких пар испытаний несколько, то вывести номера всех в порядке возрастания

Входные данные: Первое число N ($0 < N < 32767$) – количество испытаний, далее N целых значений H_n ($0 < H_n < 32767$) – результаты испытаний

Выходные данные: Построчно пары чисел – номера пар испытаний с максимальным перепадом значений.

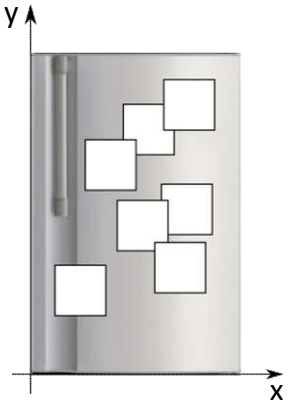
Ограничение по времени, сек.	1
Ограничение по памяти, мегабайт	1

Входные данные	Выходные данные
5 5 6 8 10 9	2 3 3 4

Возможное решение на C++

```cpp
#include <iostream>

using namespace std;

int main()
{  int n;
   cin >> n;
   int a[n+1];
   int d = 0;
   cin >> a[1];
   for ( int i = 2; i <=n; i++)
    {  cin >> a[i];
       if ( abs(a[i]-a[i-1]) > d) d = abs(a[i]-a[i-1]);
    }

   for ( int i = 2; i <=n; i++)
   if ( abs(a[i]-a[i-1]) == d) cout << i-1 << " "<< i<< endl;

    return 0;
}
```

Задача №2. Доска для записей



На пробковую доску крепятся записки. Записка представляет собой квадратный листочек со стороной **a**. Положение записки определяется координатами её верхнего левого угла в декартовой прямоугольной системе координат дверцы холодильника. Все записки для красоты имеют одинаковые размеры.

Требуется: написать программу, определяющую площадь видимой области записок.

Входные данные: Первые два целых числа, разделённых пробелом: n($0 < n < 100$) – количество записок и a ($0 < A < 10$) – длина её стороны, далее n пар чисел x, y ($0 < x, y < 65535$), разделённых пробелом – целые координаты верхнего левого угла каждой записки.

Выходные данные: одно целое число – площадь видимой области поверхности записок.

Ограничение по времени, сек.	1
Ограничение по памяти, мегабайт	1

Входные данные	Выходные данные
3 2 1 4 2 3	10

2 5	

Идея возможного решения

Воспользуемся тем, что координаты левого верхнего угла листочка и длина его стороны – целые числа. Разобьем каждый листочек на квадратики 1x1 и будем в массив записывать пары координат левого верхнего угла каждого единичного квадратика. Отсортируем все квадратики по координате x, а при равенстве координат x, по координате y. А далее посчитаем число разных квадратиков.

Решение:

```
program probl;
 type coord = record x,y: word; end;
 var fin, fout: text;
     n,d,i,j,count,kx,ky: integer;
     a: array of coord;
     b,c: coord;
BEGIN
 count := 1; SetLength(a, count);
 AssignFile(fin, 'input.txt'); reset(fin);
 readln(fin, n, d);
 for i := 1 to n do begin
//Читаем очередную координату
  readln(fin, b.x, b.y);
  if i = 1 then a[0] := b;
//Записываем в массив при условии, если такой нет
//координаты всех блоков (1x1) листочка
   for kx := 0 to d-1 do for ky := 0 to d-1 do begin
//т.к. базовая точка верхний-левый угол, то x - плюс, y - минус
    c.x := b.x + kx; c.y := b.y - ky;
    j := 0;
    while (j < count)and(a[j] <> c) do inc(j);
    if j = count then begin
      inc(count); SetLength(a, count);
      a[count-1] := c;
    end;
   end;
 end;
 CloseFile(fin);
//Вывод результата
 AssignFile(fout, 'output.txt'); rewrite(fout);
 write(fout,count);
 CloseFile(fout);
END.
```

Задача №3. S-палиндромы

Палиндром – это число, которое слева-направо и справа-налево читается одинаково Например, 121, 3223, 4 – палиндромы. Число называется s-палиндромом, если его сумма цифр является палиндромом, причём само исходное число таковым не является. Например, таким s-палиндромом является число 137, так как его сумма цифр равна 11. Число 11 – палиндром, так как читается одинаково слева-направо и наоборот. Число 22 уже не будет s-палиндромом (несмотря на то, что сумма цифр – 4 – палиндром), так как само является палиндромом. Требуется написать программу, выводящую все s-палиндромы в указанном интервале.

Входные данные

В первой строке – два натуральных числа a, b ($1 \le a, b \le 32767$) – интервал чисел для поиска s-палиндромов.

Выходные данные

Выходной файл должен содержать N строк – числа s-палиндромы в интервале [a, b] (включая a и b), либо сообщение «NO», если таковых нет.

Ограничение по времени, сек.	2
Ограничение по памяти, мегабайт	1

Пример

Входные данные	Выходные данные
820 840	821 830

Решение

Для решения задачи потребуется два фрагмента алгоритма. Первый (представлен в цикле) – предназначен для нахождения суммы цифр в числе. Второй (оформлен в виде функции pol) – определение факта принадлежности числа к палиндромам.

Таким образом, пример решения может принять следующую форму.

```
//Функция определения является ли число палиндромом
function pol(st: integer): boolean;
begin
 var s := IntToStr(st);
 result := true;
 for var i :=1 to s.Length div 2 do
  if s[i] <> s[s.Length-i+1] then begin
   result := false; break;
 end;
end;
//--- ОСНОВНАЯ ПРОГРАММА
begin
 var a,b,sm: integer;
 var n: integer = 0;
//Чтение исходных данных
```

```
 var f := OpenRead('palind.in');
 read(f,a,b); f.Close;
 f := OpenWrite('palind.out');
//Решение задачи. Перебор чисел от a до b
 for var i := a to b do begin
  sm := 0;
  if not pol(i) then begin
//Находим сумму цифр числа
   for var j := 1 to IntToStr(i).Length do
    sm += StrToInt(IntToStr(i)[j]);
   if pol(sm) then begin writeln(f,i); n += 1; end;
  end;
 end;
//Проверяем счётчик количества недопалиндромов
 if n = 0 then writeln(f,'NO');
 f.Close;
end.
```

Задача №4. Учет трафика

Сотовый оператор чтобы привлечь клиентов, каждый месяц дарит пакет не тарифицируемых минут и сообщений, взимая при этом плату за все разговоры и сообщения сверх подарка. Оператор предоставляет ежемесячный отчёт о разговорах и сообщениях, чтобы пользователь мог легко убедиться в честности оператора и правильности выставленного счёта.

Требуется помочь пользователю рассчитать требуемую для оплаты сумму.

Входные данные

В первой строке одно число N ($1 \le N \le 10^3$) – число записей о звонках и сообщениях. Вторая строка содержит два целых P_1 и P_2 и два вещественных S_1 и S_2 числа, разделённых пробелами ($1 \le P_1, P_2, S_1, S_2 \le 10^4$). P_1 и P_2 – не тарифицируемое (подарочное) количество минут и SMS соответственно. S_1 и S_2 – стоимость минуты и стоимость SMS соответственно.

Далее N строк – информация о звонках и сообщениях. Для звонка: дата, время, длительность звонка в формате MM:SS. Для сообщения: дата, время отсылки сообщения и кодовый текст «SMS». Формат даты DD.MM.YYYY, формат времени HH:MM:SS.

Выходные данные

Выходной файл должен содержать одно число – необходимая сумма оплаты с точностью до копейки.

Ограничение по времени, сек.	1

Ограничение по памяти, мегабайт	1

Пример

Входные данные	Выходные данные
7 10 3 1.20 0.53 01.11.2017 09:10:01 SMS 01.11.2017 10:03:57 SMS 02.11.2017 11:14:03 02:10 05.11.2017 13:12:56 SMS 06.11.2017 08:12:12 03:50 06.11.2017 17:55:21 04:34 30.11.2017 12:19:38 SMS	1.21

Решение

Основой при решении задачи является аккуратность в обработке предлагаемых данных. Обратим внимание на некоторые моменты. Во-первых, строк с информацией о звонке и о SMS имеют различие в длине. Во-вторых, оперировать с информацией о длительности звонков удобнее в секундах.

На языке Pascal решение может принять следующий вид:

```
var p1,p2,n,nsms,nring: integer; s1,s2,pr: real;
     s: string;
BEGIN
//Чтение исходных данных
 var f: text := OpenRead('mobil.in');
 readln(f,n); readln(f,p1,p2,s1,s2);
//Инициализация счётчиков количества SMS, длительности звонков в
//секундах
 nsms := 0; nring := 0;
//Чтение данных о звонках и SMS
 for var i: integer := 1 to n do begin
  readln(f,s);
  s := copy(s,21,length(s)-20);
  if length(s)=3 then nsms += 1
    else
     nring += StrToInt(copy(s,1,2))*60 + StrToInt(copy(s,4,2))
 end;
 CloseFile(f);
//Расчёт оплаты с учётом подаренных пакетов
 if nsms > p2 then pr := (nsms-p2) * s2;
 if nring/60 > p1 then pr += (nring-p1*60) * s1/60;
//Вывод результата
```

```
 f := OpenWrite('mobil.out');
 writeln(f, pr:1:2); CloseFile(f);
END.
```

Задача №5. Игры для умных детей

В магазине «Умный ребенок» предлагается следующая игра Полем для игры выступает полоска, разделённая на квадраты. В каждом квадрате полоски написано некоторое натуральное число. Игрок в начальный момент времени ставит фишку перед первым квадратом и может совершать следующие движения: поставить фишку на следующий квадрат или перепрыгнуть квадрат. Попадая на определённый квадрат полоски, игрок прибавляет к своему счёту число, написанное в квадрате, на который он попал. До начала игры счёт игрока равен нулю. Задача игрока «пройти» полоску, набрав минимальное количество очков.

Требуется определить минимальную сумму баллов, за которую игрок может пройти полоску.

Ограничение по времени, сек.	1
Ограничение по памяти, мегабайт	1

Формат входных данных

В первой строке одно число N ($2 \le N \le 20$) – длина полоски (количество квадратов). Во второй строке N натуральных чисел [1; 100] – последовательно значения, написанные на квадратах.

Формат выходных данных

Выходной файл должен содержать одно число – минимальная сумма баллов, за которую можно пройти игру.

Пример

Входные данные	Выходные данные
5 2 1 1 5 1	3

Решение

Один из вариантов сводится к перебору возможных путей прохода по игровой дорожке. При этом сумма баллов прохождения по этому пути

становится предметом для выявления минимального путём сравнения с предыдущим значением.
В предложенном варианте решения перебираются все теоретически возможные пути (функция NextPath), а каждый путь оценивается на соответствие условию задачи (функция Test).

На языке Pascal решение может принять следующий вид:

```
var d: array of integer; n: integer;
//Функция позволяет сгенерировать следующий маршрут движения по
//полю: 1 - клетка на которую прыгаем, 0 - клетка через которую
//прыгаем.
function NextPath(a: string): string;
begin
 for var i:integer := length(a) downto 1 do
  if a[i] = '1' then a[i] := '0' else begin
   a[i] := '1'; break;
  end;
 result := a;
end;
//Функция проверки на соответствие маршрута, описанным в задаче
//граничным условиям
function Test(s: string): boolean;
begin
 for var i: integer := 1 to length(s)-1 do
  if (s[i]='0')and(s[i+1]='0') then
    begin Result := false; exit; end;
 Result := true;
end;
//Функция определения суммы значений при прохождении
//по пути (аргумент)
function SumPath(s: string): integer;
begin
  var sm: integer := 0;
  for i: integer := 1 to length(s) do
   if s[i] = '1' then sm += d[i-1];
  result := sm;
end;

 var sm,rmin: longint;
BEGIN
 var f: text := OpenRead('game.in');
//Чтение данных из файла в массив d
 readln(f,n); SetLength(d,n);
 for var i: integer := 0 to n-1 do read(f,d[i]);
 CloseFile(f);
//Определяемся с начальным значением минимального rmin
 if n = 2 then rmin := min(d[0],d[1]) else rmin := 2000;
//Начинаем подготовку к перебору возможных путей
 if n > 2 then begin
   var st: string := ''; var stfin: string := '';
   for var i: integer := 1 to n-2 do st := st+'0';
```

```
   stfin := '00'+st; st := '01'+st;
//Начинаем перебор возможных путей
   Repeat
//Если путь соответствует условиям задачи, то ищем сумму
    if Test(st) then begin
      sm := SumPath(st);
//Выбираем минимальную сумму
      if sm < rmin then rmin := sm;
    end;
    st := NextPath(st)
   until st = stfin;
 end;
//Вывод результата
 f := OpenWrite('game.out'); write(f,rmin); CloseFile(f);
END.
```